Property-Based Interest Propagation in Ontology-Based User Model

Property-based interest propagation in ontology-based user model ⋆

Federica Cena and Silvia Likavec and Francesco Osborne

Università di Torino, Dipartimento di Informatica, Torino, Italy
{cena,likavec,osborne}@di.unito.it

Abstract. We present an approach for propagation of user interests in ontology-based user models taking into account the properties declared for the concepts in the ontology. Starting from initial user feedback on an object, we calculate user interest in this particular object and its properties and further propagate user interest to other objects in the ontology, similar or related to the initial object. The similarity and relatedness of objects depends on the number of properties they have in common and their corresponding values. The approach we propose can support finer recommendation modalities, considering the user interest in the objects, as well as in singular properties of objects in the recommendation process. We tested our approach for interest propagation with a real adaptive application and obtained an improvement with respect to IS-A-propagation of interest values.

1 Introduction

Recommender systems, both collaborative and content-based, usually suffer from cold-start and diversity problems. The *cold start* problem [20] happens at the beginning of the interaction when the system does not have enough user data to provide appropriate adaptation. The *diversity problem* [14] occurs when recommendation results, although similar to the initial object are also very similar to each other, thus lacking diversity and not providing the user with the satisfying alternatives. In the last years, several approaches have been proposed to address such problems.

Regarding the cold start problem, the most common solutions are [21]: displaying non-personalized recommendations until the user has interacted enough, asking the users directly for their interests or demographic features, clustering users in stereotypes, sharing the user models among adaptive applications [3], importing user profiles from social web applications [1]. Using ontology structure to propagate user interest values starting from a small number of initial concepts to other related concepts in the domain has proven to be a valuable tool in resolving the cold-start problem [7, 4]. Following this direction, we develop an approach for propagating user interests which enables incremental update of the user model starting from initial user feedback on domain objects.

As far as the diversity problem is concerned, the following solutions were proposed [23]: bounded greedy selection strategy where a diverse retrieval set is built starting from the concepts most similar to the initial query and choosing the additional candidates for recommendation according to their similarity *and* diversity; ordered-based

⋆ This work has been supported by PIEMONTE Project - People Interaction with Enhanced Multimodal Objects for a New Territory Experience.

retrieval where cases for recommendation are ordered based on their similarity to ideal features; compromise-driven strategy where a subset of compromises is involved. To contribute to the resolution of diversity problem, we develop an approach for propagation of user interest values based on relatedness also among distant concepts.

The main contribution of the paper is a novel algorithm for propagation of user interests to other similar and related objects which takes into account the properties of the objects in the domain and their corresponding values. Our approach helps solve the cold start problem, enabling efficient propagation even in the presence of user feedback for a small number of items. It also alleviates the diversity problem, since it allows not only propagation to similar objects, but also to related but more distant objects which share some property that the user may find desirable [23].

The approach is based on the following:

- a *semantic representation of the domain knowledge* using an OWL ontology where domain concepts are taxonomically organized, related to each other (object type properties) and enriched with data type properties;
- a methodology for calculating the *similarity (and relatedness) of domain objects* considering common properties for the objects and their corresponding values;
- a *user model* defined as an overlay on the domain ontology;
- a *strategy for building and updating the user model*, that automatically detect preferences for objects' properties starting from the user feedback.

Although our approach can be used as a preliminary step for enabling any type of recommendation, it is especially suitable for *case-based recommendation* [23], where user interest in object properties is calculated in order to select objects to recommend. However, the investigation of recommendation strategies is out of the scope of this paper.

In our previous work [7], we presented an effective approach for propagation of user interests in an ontology, following the IS-A relationships among concepts. This approach required an ontology with an explicit, well-built taxonomy of classes and subclasses. Such vertical propagation was limited to certain portions of the ontology (sub-ontologies). The approach presented in this paper is different since it is suitable for ontologies which do not have explicit and deep hierarchical structure, and where the classes are defined with restrictions on properties. Considering the properties of concepts, the propagation algorithm can reach the nodes in different sub-ontologies (hence different and sometimes distant) which would not be reachable using only IS-A propagation, allowing to solve in this way the diversity problem.

We tested this approach with a real social semantic application, WantEat [15], in order to asses the accuracy of the user model built with our approach. We also validated the advantages of our propagation mechanism, w.r.t. the vertical propagation approach presented in [7].

The rest of the paper is organized as follows. In Sect. 2, we describe the domain representation requirements, with a brief description of the treatment of properties in OWL. In Sect. 3 we describe how to calculate the similarity and relatedness between concepts in the domain ontology. We describe our user modelling approach and specific algorithm for the propagation of user interests in an ontology in Sect. 4. The results of a preliminary evaluation are given in Sect. 5. In Sect. 6 we present some related work. Finally, we conclude and give some directions for future work in Sect. 7.

2 Background: properties in OWL

To develop our approach, the domain knowledge must be represented semantically by means of ontologies expressed in OWL[1]. Ontologies represent a hierarchy of domain concepts and the features of such concepts are defined as their properties. OWL distinguishes two kinds of properties: (i) *object properties* relating objects among themselves and ii) *data type properties* relating objects to data type values. We are primarily interested in object properties, since they describe objects in terms of relations with other objects, i.e. they allow to define relations that are not IS-A. In particular, we consider OWL classes defined with restrictions on property values.

Defining classes with property restrictions. Properties can be used to define classes by means of *local anonymous classes*, i.e. collections of objects satisfying certain restrictions on certain properties. For example, `Mortadella` can be defined as a subclass of an anonymous class that has its *hasMeatKind* property restricted to `Pork`, *preparationType* property restricted to `Cooked` and *isMinced* property restricted to `Yes`. There are three ways of expressing the restrictions on the *kind* of the value[2]: (i) owl:hasValue states a specific value that the property must have; (ii) owl:allValuesFrom specifies the class of possible values the property can take (it is possible not to have any); (iii) owl:someValuesFrom specifies the class of values for at least one of the values for the property (at least one must exist). Hence, we can consider each of the concepts in our ontology, to have certain properties defined for it. These properties further describe the concepts in the ontology and can be used to calculate their mutual similarity.

Defining instance properties. The instances in the ontology inherit the properties of the classes they belong to (IS-A relation). Hence, in OWL the properties of the instances are defined by associating to each property its specific value.

3 Property-based similarity and relatedness of domain elements

Property-based similarity regards the similarity of classes defined with restrictions. For example, in an ontology describing cold cuts, certain concepts can have these properties: *hasMeatKind*, *preparationType*, *isMinced*. Consider the following cold cuts from this ontology: `Mortadella`, `Cooked_Ham` and `Raw_Ham`. `Mortadella` is made with pork, is cooked and minced, `Cooked_Ham` is made with pork, is cooked and not minced and `Raw_Ham` is made with pork, is not cooked and not minced. If we simply count the properties which certain cold cuts have in common, we see that `Mortadella` is more similar to `Cooked_Ham` than to `Raw_Ham`, since `Mortadella` and `Cooked_Ham` have two properties in common, whereas `Mortadella` and `Raw_Ham` have only one property in common. But, it can happen that for a certain property, two values are given. For example, for `Salame_Pavese`, the property *hasAddedMeat* has two values: veal and chicken.

There are three kinds of restriction declarations used to define properties for certain classes: (i) owl:hasValue; (ii) owl:allValuesFrom; (iii) owl:someValuesFrom. When calculating the property-based similarity of two domain elements N_1 and N_2 [3], we start

[1] `http://www.w3.org/TR/owl-ref`

[2] It also is possible to express *cardinality restrictions* on the property, by using minCardinality, maxCardinality and cardinality. We are not dealing with cardinality restrictions here.

[3] We consider equal the properties defined with EquivalentProperty.

from Tversky's feature-based model of similarity [24], where similarity between objects is a function of both their common and distinctive characteristics:

$$\textsc{sim}_T(N_1, N_2) = \frac{\alpha(\psi(N_1) \cap \psi(N_2))}{\beta(\psi(N_1) \setminus \psi(N_2)) + \gamma(\psi(N_2) \setminus \psi(N_1)) + \alpha(\psi(N_1) \cap \psi(N_2))}$$

where $\psi(N)$ is the function describing all the relevant features of N, and $\alpha, \beta, \gamma \in \mathbb{R}$ are parameters permitting to treat differently various components. By taking $\alpha = 1$ we obtain maximal importance of the common features of the two concepts and by taking $\beta = \gamma$ we obtain non-directional similarity measure. We will use $\alpha = 1$ and $\beta = \gamma = 1$.

So we have to calculate

- *common features of N_1 and N_2:* $\textsc{cf}(N_1, N_2) = \psi(N_1) \cap \psi(N_2)$,
- *distinctive features of N_1:* $\textsc{df}(N_1) = \psi(N_1) \setminus \psi(N_2)$ and
- *distinctive features of N_2:* $\textsc{df}(N_2) = \psi(N_2) \setminus \psi(N_1)$.

To this aim, for each property p, we calculate $\textsc{cf}_p$, $\textsc{df}^1_p$ and $\textsc{df}^2_p$, which denote how much the property p contributes to common features of N_1 and N_2, distinctive features of N_1 and distinctive features of N_2, respectively. We distinguish the following six different cases based on how the restrictions on properties are defined for N_1 and N_2:

1. The property p is defined with owl:hasValue in N_1 and N_2. If p has h' different values in N_1 and h'' different values in N_2, and we denote by k the number of times P_1 and P_2 have the same value for p, then $\textsc{cf}_p = \frac{k^2}{h'h''}$ $\textsc{df}^1_p = \frac{h'-k}{h'}$ $\textsc{df}^2_p = \frac{h''-k}{h''}$.
2. The property q is defined in N_1 with ⟨owl:allValuesFrom rdf:resource="#A_1"⟩ at most once and in N_2 with ⟨owl:allValuesFrom rdf:resource="#A_2"⟩ at most once. Let a_1 (resp. a_2) be the number of sub-classes of A_1 (resp. A_2). If A_1 and A_2 are equivalent or equal, $a_1 = a_2$ and $\textsc{cf}_q = \frac{1}{(a_1+1)^2}$. Otherwise $\textsc{df}^1_q = \frac{1}{a_1+1}$ and $\textsc{df}^2_q = \frac{1}{a_2+1}$.
3. The property r is defined in N_1 ⟨owl:someValuesFrom rdf:resource="#S_1"⟩ at mst once and in N_2 with ⟨owl:someValuesFrom rdf:resource="#S_2"⟩ at most once. Let s_1 (resp. s_2) be the number of sub-classes of S_1 (resp. S_2) and w be the number of classes in the whole domain. If S_1 and S_2 are equivalent or equal, $s_1 = s_2$ and $\textsc{cf}_r = \frac{1}{(s_1+1)^2 w^2}$. Otherwise $\textsc{df}^1_r = \frac{1}{(s_1+1)w}$ and $\textsc{df}^2_r = \frac{1}{(s_2+1)w}$.
4. The property t is defined m times in N_1 with owl:hasValue and once in N_2 with ⟨owl:allValuesFrom rdf:resource="#A_3"⟩. If a_3 is the number of sub-classes of A_3, then $\textsc{cf}_t = \frac{1}{m(a_3+1)}$, $\textsc{df}^1_t = \frac{m-1}{m}$ and $\textsc{df}^2_t = \frac{1}{a_3+1}$.
5. The property x is defined n times in N_1 with owl:hasValue and once in N_2 with ⟨owl:someValuesFrom rdf:resource="#S_3"⟩. If s_3 is the number of sub-classes of S_3 and w is the number of classes in the whole domain then $\textsc{cf}_x = \frac{1}{n(s_3+1)w}$, $\textsc{df}^1_x = \frac{n-1}{n}$ and $\textsc{df}^2_x = \frac{1}{(s_3+1)w}$.
6. The property y is defined once with ⟨owl:allValuesFrom rdf:resource="#A_4""⟩ in N_1 and once with ⟨owl:someValuesFrom rdf:resource="#S_4""⟩ in N_2. Let a_4 (resp. s_4) be the number of sub-classes of A_4 (resp. S_4) and w be the number of classes in the whole domain. Then $\textsc{cf}_y = \frac{1}{(a_4+1)(s_4+1)w}$, $\textsc{df}^1_y = \frac{1}{a_4+1}$ and $\textsc{df}^2_y = \frac{1}{(s_4+1)w}$.

Finally, to calculate all common and distinctive features of N_1 and N_2 we repeat the above process for each property defined for N_1 and N_2, obtaining:

$$\textsc{cf}(N_1, N_2) = \Sigma_{i_p=1}^{n_p} \textsc{cf}_{p_{i_p}} + \Sigma_{i_q=1}^{n_q} \textsc{cf}_{q_{i_q}} + \Sigma_{i_r=1}^{n_r} \textsc{cf}_{r_{i_r}} + \Sigma_{i_t=1}^{n_t} \textsc{cf}_{t_{i_t}} + \Sigma_{i_x=1}^{n_x} \textsc{cf}_{x_{i_x}} + \Sigma_{i_y=1}^{n_y} \textsc{cf}_{y_{i_y}}$$

$$\textsc{df}(N_1) = \Sigma_{i_p=1}^{n_p} \textsc{df}^1_{p_{i_p}} + \Sigma_{i_q=1}^{n_q} \textsc{df}^1_{q_{i_q}} + \Sigma_{i_r=1}^{n_r} \textsc{df}^1_{r_{i_r}} + \Sigma_{i_t=1}^{n_t} \textsc{df}^1_{t_{i_t}} + \Sigma_{i_x=1}^{n_x} \textsc{df}^1_{x_{i_x}} + \Sigma_{i_y=1}^{n_y} \textsc{df}^1_{y_{i_y}}$$

$$\textsc{df}(N_2) = \Sigma_{i_p=1}^{n_p} \textsc{df}^2_{p_{i_p}} + \Sigma_{i_q=1}^{n_q} \textsc{df}^2_{q_{i_q}} + \Sigma_{i_r=1}^{n_r} \textsc{df}^2_{r_{i_r}} + \Sigma_{i_t=1}^{n_t} \textsc{df}^2_{t_{i_t}} + \Sigma_{i_x=1}^{n_x} \textsc{df}^2_{x_{i_x}} + \Sigma_{i_y=1}^{n_y} \textsc{df}^2_{y_{i_y}}.$$

where n_p (resp. n_q, n_r, n_t, n_x and n_y) is the number of properties defined in each of six possible ways. Finally, we calculate the similarity between two entities N_1 and N_2 defined with restrictions as follows:

$$\text{SIM}(N_1, N_2) = \frac{\text{CF}(N_1, N_2)}{\text{DF}(N_1) + \text{DF}(N_2) + \text{CF}(N_1, N_2)}.$$

Not all the features have the same importance in defining a concept. For example, it is possible to account for relevance of properties by providing the relevance factors (either as a-priori expert values or as user preferences) $R^p_{i_p}, i_p = 1, \ldots, n_p$, for each property p (and analogously for all the others). In this way, some properties become more important than the others, e.g. in case of cold cuts ontology *hasMeatKind* can be considered more important than *isSpicy*. In this case the formula for calculating the mutual similarity between domain items N_1 and N_2 becomes:

$$\text{SIM}^r(N_1, N_2) = \frac{\text{CF}^r(N_1, N_2)}{\text{DF}^r(N_1) + \text{DF}^r(N_2) + \text{CF}^r(N_1, N_2)}$$

where $\text{CF}^r(N_1, N_2) = \Sigma_{i_p=1}^{n_p} R_{i_p} \text{CF}_{p_{i_p}} + \ldots + \Sigma_{i_y=1}^{n_y} R_{i_y} \text{CF}_{y_{i_y}}$ and similarly for $\text{DF}^r(N_1)$ and $\text{DF}^r(N_2)$.

Another feature we want to take into account is the presence of equivalent classes, even though they are not defined as restrictions. We assume that two classes declared equivalent with equivalentClass would have similarity based on properties equal to 1.

As far as individuals are concerned (instances of the classes) we simply compare the values-property pairs declared for each instance. This is a simple case, analogous to the first case in the above discussion.

As opposed to similarity, which finds the elements similar to each other, relatedness helps find the elements that are *related* to each other. For example, if a certain product is *producedBy* a certain company and another product is *soldBy* the same company, these two products might not be similar but are definitely related. Finding related elements in the domain, permits us to cover different sub-ontologies of the domain, which are not reachable only with similarity, hence helping to resolve the diversity problem.

In order to calculate the property-based relatedness of domain elements we apply similar reasoning as for calculating similarity, but considering the values that are the same, for the different properties having the same ancestor. More precisely, considering completely unrelated properties would lead to relating very different elements and that is not our goal. Instead, we choose the properties that are in some way related (for example all are descendants of the same class) and calculate relatedness based on these properties. Lack of space does not allow us to go into more details on relatedness.

4 User model

In this section we describe our approach to model users. We start with the definition of the user model (Sect. 4.1), followed by the description of the user feedback and how we use it (Sect. 4.2), to conclude with our technique for user model update and interest propagation (Sect. 4.3).

4.1 User model definition

As described above, the domain is represented by means of an OWL ontology, with the explicit specification of the concepts' properties. The user model is defined as an

overlay on such an ontology (*ontology-based user model*), in order to represent the user interests for the domain concepts. More precisely, each ontological user profile is an instance of the domain ontology, where each node in the ontology has an interest value associated to it. This means that each node N in the domain ontology can be seen as a pair $\langle N, \mathcal{I}(N)\rangle$, where $\mathcal{I}(N)$ is the interest value associated to node N. Hence, the user model contains the values of user interests for the concepts in the ontology. Notice that at the beginning of the interaction the model is empty and the interest values will be inserted in the further update phases. The user model contains not only the information about the user's interest in domain objects, but also in all the objects' properties.

4.2 User feedback

We chose not to directly ask the users about their preferences, but to automatically detect user preferences for properties according to users' behavior. The system records implicitly the user actions, inferring from them the interest for the object the action is performed on, and uses it to incrementally create and update the user model by modifying the interest values for certain domain objects.

Action	**Weight**
Bookmarking an object	0.9
Tagging an object	0.7
Commenting on an object	0.5
Selecting an object	0.3
Rating an object	0.1*vote

Table 1. Weights associated to user actions

Following Kobsa [11], each type of user feedback can be a signal of different user interest and as such can have different impact on the user model. We consider 5 possible most common typologies of generic user feedback in an adaptive social system[4]: (i) selecting an object, (ii) tagging, (iii) commenting, (iv) rating/voting on $1-5$ scale, or (v) putting an item into favorites/bookmarking it. Each of these actions is assigned a certain weight f, following the approach in [6] (see Table 1). These values are registered in the log files and analyzed further to calculate the user interest in both, the objects receiving the feedback and in their properties, as described in Sect. 4.3.

4.3 User model update and interest propagation

Adapting the approach in [7], each time a user provides a feedback, we first calculate the user *direct interest* $\mathcal{I}_{\mathrm{D}}(N)$ for the node N receiving the direct feedback as a weighted sum of old interest and sensed interest: $\mathcal{I}_{\mathrm{D}}(N) = \sigma_1\mathcal{I}_{\mathrm{O}}(N) + \sigma_2\mathcal{I}_{\mathrm{S}}(N)$, $\sigma_1, \sigma_2 \in \mathbb{R}$ such that $\sigma_1 + \sigma_2 = 1$. **Old interest** $\mathcal{I}_{\mathrm{O}}$ is the old value for the user interest (initially equal to zero). The **sensed interest** $\mathcal{I}_{\mathrm{S}}$ is the value obtained from the direct feedback of the user. It depends on the user feedback for the node and the position of the node in the ontology, since the nodes lower down in the ontology represent specific concepts, and as such signal more precise interest than the nodes represented by upper classes in the ontology, expressing more general concepts. In order to calculate the interest sensed by a given node N, we use the sigmoid function

$$\mathcal{I}_{\mathrm{S}}(N) = \frac{l(N)}{\mathrm{MAX}(1 + e^{-\mathsf{f}(N)})}$$

[4] The choice of actions to consider depends on a particular domain and application being used.

where $l(N)$ is the level of the node receiving the feedback, MAX is the level of the deepest node in the ontology and $\mathrm{f}(N)$ is the feedback obtained from the user for the node N.

In our previous work [7] we used a power law with a negative exponent to weight the feedbacks. We now use the sigmoid function since it allows negative feedbacks, moves between -1 and 1 and does not need any constant.

In addition to being used for updating the direct interest value for the domain objects, sensed interest is also used to update the interest values for the object properties in the user model. Let $N_1, \ldots, N_k$ be the nodes that have a certain property p defined for them and each of them has $p_1, \ldots, p_{n_k}, n_k \in \mathbb{N}$ total properties defined. If the corresponding sensed interest values are $\mathcal{I}_{\mathrm{S}}(N_1), \ldots, \mathcal{I}_{\mathrm{S}}(N_k)$, then the interest value for the given property p is calculated as follows:

$$\mathcal{I}(p) = \frac{1}{k} \Sigma_{i=1}^{k} \frac{\mathcal{I}_{\mathrm{S}}(N_i)}{n_i}.$$

If the relevance for the properties is provided in the system, then the above interest value is multiplied by the relevance value for that property.

Moreover, this value is also used for the subsequent propagation phase, in which the *inferred* interest values for the similar objects can be calculated as: $\mathcal{I}_{\mathrm{I}}(M) = \pi_1 \mathcal{I}_{\mathrm{O}}(M) + \pi_2 \mathcal{I}_{\mathrm{P}}(N, M)$ where **propagated interest** $\mathcal{I}_{\mathrm{P}}$ is the value obtained by property-based propagation and $\pi_1, \pi_2 \in \mathbb{R}$, such that $\pi_1 + \pi_2 = 1$. We use the properties of each of the domain elements to calculate their mutual similarity (see Sect. 3) and decide to which elements to propagate the user interest. This propagation does not have any particular direction (as opposed to the one described in [7]) and permits us to propagate the user interests to various (sometimes quite distant) nodes of the ontology. The value of propagated interest value is calculated using the hyperbolic tangent function as follows:

$$\mathcal{I}_{\mathrm{P}}(N, M) = \frac{e^{2\mathrm{SIM}(N,M)} - 1}{e^{2\mathrm{SIM}(N,M)} + 1} \mathcal{I}_{\mathrm{S}}(N)$$

where $\mathcal{I}_{\mathrm{S}}(N)$ is the sensed interest of the node N receiving the feedback and $\mathrm{SIM}(N, M)$ is the *similarity* between the node N receiving the feedback and the node M receiving the propagated interest (see Sect. 3). It is possible to propagate the user interest also to related nodes, using relatedness instead of similarity. We keep these two interest values, $\mathcal{I}_{\mathrm{D}}$ and $\mathcal{I}_{\mathrm{I}}$ separated for each concept, but the total interest for each concepts is obtained as: $\mathcal{I}(N) = \epsilon_D \mathcal{I}_{\mathrm{D}} + \epsilon_I \mathcal{I}_{\mathrm{I}}$, where $\epsilon_D, \epsilon_I \in \mathbb{R}$ and $\epsilon_D + \epsilon_I = 1$. By varying the constants ϵ_1 and ϵ_2 it is possible to assign different level of importance to either $\mathcal{I}_{\mathrm{D}}$ or $\mathcal{I}_{\mathrm{I}}$.

5 Evaluation

We evaluated our approach in *WantEat* application[15], where food such as cold cuts and wines, but also shops, restaurants etc, are intelligent objects able to provide recommendations. The domain is represented using a set of OWL ontologies, and the user model is defined as an overlay on such ontologies. For the experimental evaluation we selected the portion of the ontology regarding cold cuts (see Fig. 1), since it is fairly well balanced and easy for users to provide feedback. The system shows an adaptive behavior, ordering the objects returned by a search or displayed during the navigation according to user interest.

Hypothesis and Experimental Design. We assumed that our algorithm can help us generate ordered lists of objects having good correlation with the user preferences. Fur-

thermore, we wanted to compare the present *property-based* propagation approach with the *vertical* propagation approach developed in [7] where the propagation is based on the lengths of the edges between classes.

We designed a questionnaire to collect users' preferences regarding cold cuts. In the first part, labelled U, the users voted for 8 non-leaf classes (e.g. `Minced_ Cooked_ ColdCuts`) on 1-10 scale. In the second part, labelled L, the users chose 4 leaf classes (e.g. `Wurstel`, `Speck` etc.) they "like very much" and 4 leaf classes they "like enough" among the total of 15 leaf classes.

We initially started with 100 subjects, 19-45 years old, recruited according to an availability sampling strategy[5]. Later, we restricted the initial sample to 87 users, eliminating all the users who assigned the same vote to 5 out of 7 classes, this being a strong indicator of random preferences assignment.

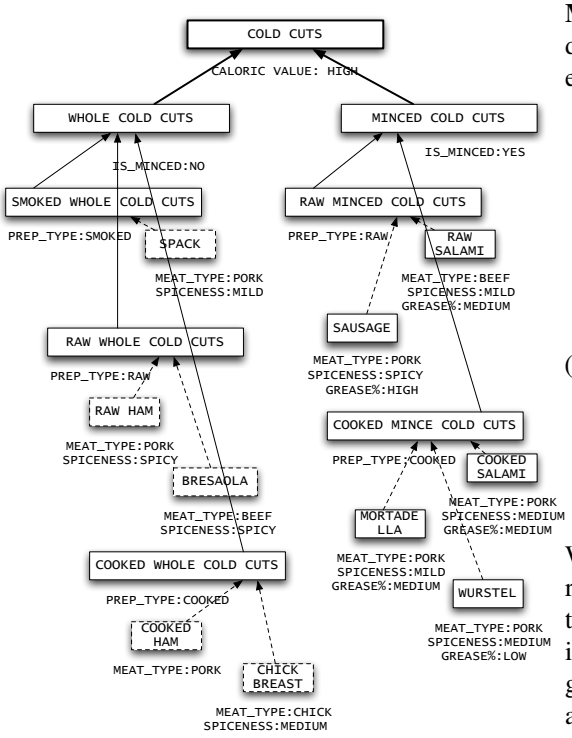

Fig. 1. The part of domain ontology describing cold cuts

Measures and Material. We distinguished two phases in the evaluation process:

(i) *Exhaustive propagation evaluation*, where we generated an ordered list of both upper and leaf classes starting from one half of the two lists generated by the user and comparing the generated list with the remaining half.

(ii) *Upward propagation evaluation*, where we generated an ordered list of upper classes using the list L and we compared them with the classes in the list U.

We used Spearman's rank correlation coefficient ρ to compute the association between the original user's list and the algorithm generated list, since it allows to address the possible ties in the ordered lists. $\rho = 1$ occurs when no repeated values exist and the two list are in the same order, $\rho \geq 0.5$ points to a "fair direct correlation", $\rho \geq 0.7$ to a "good direct correlation" and $\rho \geq 0.9$ to a "strong direct correlation".

Results. We computed the Spearman's coefficient ρ for 87 pairs of lists using both the property based propagation (red line) and the vertical propagation (dashed blue line) [7].

Exhaustive propagation. Using the property-based propagation technique we obtained the results shown in Figure 2. In 90% of the cases we acquired a list with a

[5] Much research in social science is based on samples obtained through non-random selection, such as the availability sampling, i.e. a sampling of convenience, based on subjects available to the researcher, often used when the population source is not completely defined.

positive association with the user preferences. More impressive, in 25% of the cases we computed a list with a good positive association ($\rho \geq 0.7$). Only in 2% of the cases we obtained a list with a moderate inverse correlation ($\rho \leq -0.5$). On the other hand, applying the vertical propagation yields the results not much better then random. In fact, the ability to propagate interests also horizontally is required.

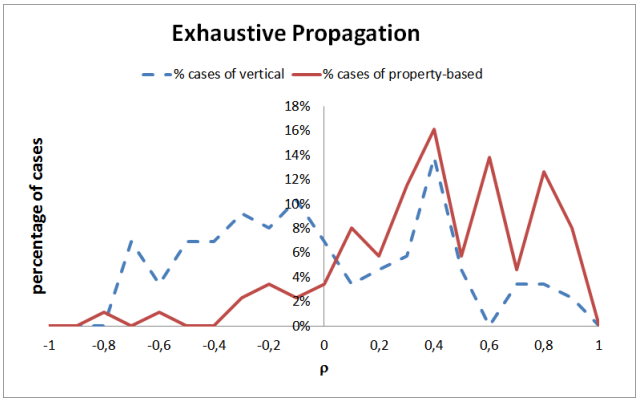

Fig. 2. Distribution of cases for various values of ρ for the exhaustive propagation.

Upward propagation. Figure 3 shows the distribution of the values of ρ for the 87 pair of lists compared. We can see that in 86% of the cases with the *property-based* propagation and in 62% of the cases with the *vertical* propagation we generated a list with a positive association with the user preferences. Therefore, the flexibility of property-based propagation allows for a better overall performance. Moreover, after only providing the feedback eight times, we are able to obtain a "good correlation" ($\rho \geq 0.7$) in 28% of the cases with property-based propagation, as opposed to 12% for the vertical propagation. The number of misleading cases with $\rho \leq -0.5$ is 1% for the property-based propagation and 13% for the vertical one.

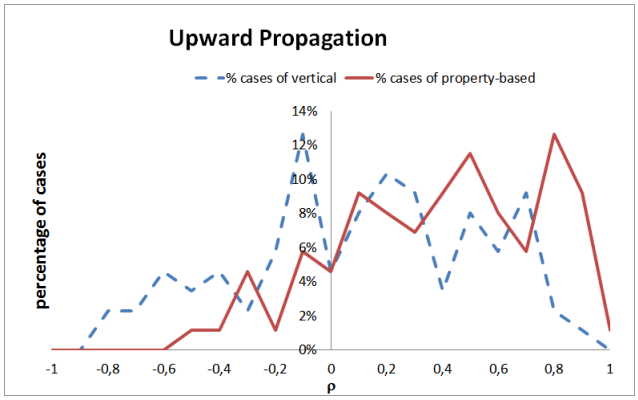

Fig. 3. Distribution of cases for various values of ρ for the upward propagation. A comparison between the *property-based* and the *vertical* approach.

Discussion. The results show that the *property-based* approach works well, both for he exhaustive and upward propagation. It appears to perform better then *vertical* tech-

nique since it allows for the exhaustive propagation and it also yields better result in the upward case. We were able to generate lists with a good correlation ($\rho \geq 0.7$) in more then one forth of the cases after small amount of feedback. The risk of suggesting a misleading list to the user is also very low for both tests. Therefore, our technique, while learning the user preferences very quickly, rarely misfires or introduces anomalies in the user model. Thus it can safely be applied alone or in combination with other methods to address the cold start and the diversity problems.

6 Related work

Since our approach requires the use of ontologies to represent domain knowledge and the user model is represented as an overlay on such ontologies, it is similar to *ontology-based recommender* [13, 22, 5, 9]. Similarly to these works we take advantage of the enhanced semantics representation, and user profiles are compared at a finer level than in usual recommender systems. What is different in our approach is the way we compute item-item similarity based on properties. [13] and [22], in order to update the interest values in the ontology, exploit only IS-A relationships. In the approach in [5], the concept, item, and user spaces are clustered in a coordinated way, and the resulting clusters are used to find similarities among individuals at multiple semantic layers. [9] take into account the semantic relatedness between different concepts in terms of semantic words relations (synonymy, hyponymy, and meronymy).

Notice that in this paper we use the term "ontology-based user model" in a different sense with respect to other works [8], that define ad hoc ontology to represent the user features in the user model. Differently, we only use the ontology to model the domain, and the user model is defined as an overlay over the domain ontology, as it has been done for a long time in educational systems [4, 18]

In similar fashion to us, other approaches make use of ontological structure to calculate similarity among concepts. In addition to Tversky's feature-based model of similarity [24], semantic similarity can be calculated in two general modes. Resnik's notion of semantic similarity [19] is based on information content in an IS-A taxonomy, given by the negative logarithm of the probability of occurrence of the class in a text corpus. The closest class subsuming compared concepts provides the shared information for both and gives the measure of their similarity. Rada et al. [17] use the ontology graph structure, using the distance between nodes (the number of edges or the number of nodes between the two nodes) to calculate the similarity. These three basic measures of similarity gave rise to many combined approaches. In Jiang and Conrath [10] distance based approach is improved with the information content one. The semantic similarity introduced by Pirró and Euzenat in [16] combines the feature-based model of similarity with the information theoretical one, where Lin [12] introduces an information-theoretic definition of similarity based on a set of assumptions about similarity, calculated as the ratio between the amount of information that two concepts have in common and the amount of information needed to fully describe them. Smyth [23] takes into account individual features of concepts and each feature has its own similarity function defined for it. Similarly to us, they also introduce the weights which help distinguish the importance of individual features when calculating similarity.

7 Conclusions and future work

This paper presents a novel approach for the propagation of user interest values in a domain ontology, considering similarity (and relatedness) of the domain concepts, based on their properties values. This approach advances the current state of the art by supporting finer recommendation modalities (e.g. case-based or item-based recommendations), since it is able to take user interests in object properties into account, thus calculating similarity among objects in a more precise way. The evaluation of the approach in gastronomic domain proved its effectiveness in improving the user model accuracy.

Notice that, when calculating similarity, we consider the values of the concept's properties, both data type and object type ones. In the first case, we do not consider the values represented as textual descriptions. In case of object-type properties, we only take into account the presence or absence of the same values of properties. Further possible development would be to give a priori similarity values between properties (e.g. soft cheese would be more similar to medium than to hard cheese). We also do not take into account cardinality restrictions, as well as restrictions defined as intersections, unions and complements, leaving these aspects for future work.

In addition, we intend to combine our approach with the vertical propagation of user interests presented in [7], obtaining a complete approach for dealing with missing user interest values. Also, it should be possible to extend the propagation of interests also to similar users. Furthermore, we want to make the user model more dynamic by taking in consideration the timing of the feedbacks and the context. We aim also at adding a confidence measure to propagation, i.e. to know how much the inference of user interest is reliable. Finally, we intend to test the user model created with our approach with different recommendation algorithms which take user preferences for properties into account to calculate recommendations, such as *case-based recommendation* [23]. In this way, we would be able to recommend restaurants and shops, based on user feedback on food products, starting from food features. Our approach could be also used to provide *multi-criteria item-based collaborative filtering recommendation* [2] where recommendations are computed by finding items similar to the other items the user likes.

References

1. F. Abel, S. Araújo, Q. Gao, and G.-J. Houben. Analyzing cross-system user modeling on the social web. In *Web Engineering - 11th International Conference, ICWE 2011*, volume 6757 of *LNCS*, pages 28–43. Springer, 2011.
2. G. Adomavicius, N. Manouselis, and Y. Kwon. Multi-criteria recommender systems. In *Recommender Systems Handbook*, pages 769–803. 2011.
3. L. Aroyo, P. Dolog, G.-J. Houben, M. Kravcik, A. Naeve, M. Nilsson, and F. Wild. Interoperability in personalized adaptive learning. *Educational Technology & Society*, 9(2):4–18, 2006.
4. P. Brusilovsky and E. Millán. User models for adaptive hypermedia and adaptive educational systems. In *The Adaptive Web, Methods and Strategies of Web Personalization*, volume 4321 of *LNCS*, pages 3–53. Springer, 2007.
5. I. Cantador, A. Bellogín, and P. Castells. A multilayer ontology-based hybrid recommendation model. *AI Communications*, 21(2-3):203–210, 2008.

6. F. Carmagnola, F. Cena, L. Console, O. Cortassa, C. Gena, A. Goy, I. Torre, A. Toso, and F. Vernero. Tag-based user modeling for social multi-device adaptive guides. *User Modeling and User-Adapted Interaction*, 18:497–538, 2008.
7. F. Cena, S. Likavec, and F. Osborne. Propagating user interests in ontology-based user model. In *Advances in Artificial Intelligence, AI*IA '11*, volume 6934 of *LNCS*, pages 299–311. Springer-Verlag, 2011.
8. D. Heckmann, T. Schwartz, B. Brandherm, M. Schmitz, and M. von Wilamowitz-Moellendorff. GUMO - the General User Model Ontology. In *10th Int. Conference on User Modeling, UM '05*, volume 3538 of *LNCS*, pages 428–432. Springer, 2005.
9. W. IJntema, F. Goossen, F. Frasincar, and F. Hogenboom. Ontology-based news recommendation. In *2010 EDBT/ICDT Workshops*, ACM Int. Conf. Proc. Series. ACM, 2010.
10. J. Jiang and D. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. In *International Conference on Research in Computational Linguistics*, pages 19–33, 1997.
11. A. Kobsa, J. Koenemann, and W. Pohl. Personalized hypermedia presentation techniques for improving online customer relationship. *The Knowledge Engineering Review*, 16(2):111–155, 2001.
12. D. Lin. An information-theoretic definition of similarity. In *15th International Conference on Machine Learning ICML '98*, pages 296–304. Morgan Kaufmann Publishers Inc., 1998.
13. S. E. Middleton, N. R. Shadbolt, and D. C. De Roure. Ontological user profiling in recommender systems. *ACM Transactions on Information Systems*, 22:54–88, 2004.
14. D. O'Sullivan, B. Smyth, and D. C. Wilson. Preserving recommender accuracy and diversity in sparse datasets. *Int. Journal on Artificial Intelligence Tools*, 13(1):219–235, 2004.
15. PIEMONTE Team. Interacting with a social web of smart objects for enhancing tourist experiences. In *ENTER 2012 conference, Helsingborg*, 2012.
16. G. Pirrò and J. Euzenat. A feature and information theoretic framework for semantic similarity and relatedness. In *9th International Semantic Web Conference, ISWC '10*, volume 6496 of *LNCS*, pages 615–630. Springer, 2010.
17. R. Rada, H. Mili, E. Bicknell, and M. Blettner. Development and application of a metric on semantic nets. *IEEE Trans. on Systems Management and Cybernetics*, 19(1):17–30, 1989.
18. L. Razmerita, A. Angehrn, and A. Maedche. Ontology-based user modeling for knowledge management systems. In *9th Int. Conference on User modeling, UM '03*, volume 2702 of *LNCS*, pages 213–217. Springer, 2003.
19. P. Resnik. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *Journal of Artificial Intelligence Research*, 11:95–130, 1999.
20. G. Salton and M. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill Book Company, 1984.
21. A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock. Methods and metrics for cold-start recommendations. In *25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '02*, pages 253–260. ACM, 2002.
22. A. Sieg, B. Mobasher, and R. Burke. Web search personalization with ontological user profiles. In *16th ACM Conference on Information and Knowledge Management*, CIKM '07, pages 525–534. ACM, 2007.
23. B. Smyth. Case-based recommendation. In *The Adaptive Web, Methods and Strategies of Web Personalization*, volume 4321 of *LNCS*, pages 342–376. Springer, 2007.
24. A. Tversky. Features of similarity. *Psychological Review*, 84(4):327–352, 1977.